Capítulo IV

Experimento base

El experimento de reconocimiento presentado en esta sección está basado en modelos acústicos independientes del contexto. Los experimentos independientes del contexto proveen un experimento base estándar sobre el cual realizar mejoras subsecuentes, como incorporar información contextual. Para refinar el diseño del sistema se entrenó en un conjunto de 270 locutores del corpus de *Tlatoa Common Questions* y la evaluación inicial se llevó a cabo usando otro subconjunto del corpus de 70 locutores. Como ya se mencionó anteriormente, estos conjuntos no incluyen la frase de habla espontánea o *stories*.

El experimento presentado en este capítulo utiliza modelos acústicos independientes del contexto. Estos modelos están construidos siguiendo la metodología del CSLU *toolkit,* incluyendo el uso de *Mel Frecuency Cepstral Coefficients* (MFCC´s).

4.1 Conjunto de fonemas del vocabulario.

Como vimos anteriormente, los fonemas constituyen las unidades básicas de un lenguaje. Esto significa que la pronunciación de una palabra se puede describir como una secuencia de fonemas. Para algunos autores el español consta de un conjunto de 37 fonemas [Lander96]. Para realizar este experimento se tomaron todos excepto algunos de ellos que no se aplican para el caso del español hablado en México. A continuación se muestra una tabla que contiene todos estos fonemas.

Fonema	Posibles alófonos
a_f	a | a_h | a_fp | a_x
b_f	bc b | V
d_f	dc d | D
e_f	e | e_h | e_fp | e_x
f_f	f
g_f	gc g | G
i_f	i | i_h | i_fp | i_x
j_f	x | x_fp
k_f	kc k
l_f	l
ll_f	dZc dZ
m_f	m | m_fp
n_f	n | N
nj_f	nj
o_f	o | o_h | o_x
p_f	pc p
r_f	r | rr | rZ
s_f	s | s_v
t_f	tc t
CH_f	tSc tS
u_f	u | u_h | u_fp | u_x | w
ia_f	ia
ie_f	ie
io_f	io
ua_f	wa
ue_f	we

Tabla 4.1 Conjunto de fonemas del vocabulario

La tabla 4.1 contiene dos columnas: la primera columna de izquierda a derecha corresponde a las unidades fonéticas que se manejaron en este proyecto. La siguiente columna corresponde a los posibles alófonos con que puede contar cada una de estas unidades. En la segunda columna las unidades están a un nivel acústico más bajo. Por ejemplo, {tc t} representa el oclusivo /t/ precedido por su *closure*, dos partes que se mapean a una sola unidad fonética que se denota como “t fonema”, ó t_f. Por otro lado, en el caso de g_f, d_f, b_f , se utiliza el caracter “ | ” en la segunda columna para indicar que estas unidades tienen varios alófonos.

4.2 Distribución de los datos

Siguiendo la metodología del *toolkit* el proceso de reconocimiento de voz consta de tres fases principales:

- Fase de entrenamiento
- Fase de desarrollo
- Fase de evaluación

Por eso es necesario dividir los datos que se obtuvieron para formar el corpus de teléfono en tres grupos: uno para entrenamiento, otro para desarrollo, y el último para la prueba final.

El primer conjunto de datos se utiliza para entrenar la red neuronal. Es importante que se tengan suficientes muestras de cada fonema para asegurar un buen modelado de éstos y por tanto un mejor reconocimiento. Los datos para la etapa de desarrollo se emplean para evaluar el nivel de reconocimiento de la red en cada iteración. Es necesario que este conjunto de datos sea diferente al de entrenamiento porque se requiere de una generalización; es decir, se espera que el reconocedor obtenga un desempeño adecuado para datos diferentes a los del conjunto de entrenamiento.

Finalmente, en la etapa de evaluación se evalúa el desempeño de la red neuronal que haya obtenido el mayor porcentaje de reconocimiento. Es necesario que los datos dentro de este conjunto sean totalmente desconocidos para la red.

4.3 Otras investigaciones similares

A continuación se presenta una tabla donde se muestran investigaciones en reconocimiento fonético para el idioma inglés, con el propósito de darse una idea de en que tanto cae la media del nivel de reconocimiento fonético actual.

Experimento	**Investigador**	**Conjunto Prueba**	**Año**	**Precisión**
REPN	Robinson	NIST Core	1992	73.9%
HMM	Lamel & Gauvain	NIST Core	1993	69.1%
SUMMIT	Philips & Glass	NIST Core	1994	68.5%
SSM	Ostendor & Roucos	BU Test	1990	66.7%
HMM	Lee	KFL Test	1989	66.1%
Dynamical System	Digilakis	BU Test	1992	63%

Tabla 4.2 Resultados de precisión en reconocimiento fonético para varios experimentos usando el corpus de voz TIMIT.

En muchos casos los resultados no son directamente comparables a partir del hecho de que se emplearon diferentes conjuntos prueba y diferentes niveles de complejidad en los modelos (por ejemplo, algunos resultados están basados en modelos independientes del contexto y otros están basados en modelos dependientes del contexto). Los resultados citados reflejan las mejores precisiones reportadas actualmente. Muchos de los experimentos descritos arriba han sido usados para generar clasificación fonética y resultados de reconocimiento usando el corpus TIMIT. Por esto ello provee un marco de referencia.

Cabe mencionar que los resultados que se obtengan en este trabajo no se pueden comparar directamente con los que se mencionan en esta sección, debido a tres razones principalmente:

- El corpus de evaluación no es TIMIT.
- Los datos que se ocupan en esta tesis son grabados por teléfono, lo que hace más difícil el reconocimiento en este trabajo. TIMIT consiste en datos grabados por micrófono de alta calidad en ambientes de bajo ruido.
- El idioma Español tiene menos fonemas que el idioma Inglés, lo que hace hasta cierto punto más fácil el reconocimiento aquí presentado.

4.4. Tcl y Tk

Tcl y Tk proveen un ambiente de programación para desarrollar y usar aplicaciones gráficas. Tcl es un script para el control y extensión de aplicaciones; su nombre viene de “tool command language”. Tcl provee facilidades de programación generales, como variables ciclos y procedimientos, que son muy útiles para una gran variedad de aplicaciones. Su intérprete utiliza una librería de procedimientos de C que pueden ser incorporados facilmente en aplicaciones, y cada aplicación puede extender las características centrales de Tcl [Ousterhout94].

Para la implantación de las herramientas que permiten la evaluación y el desarrollo de un sistema de reconocimiento fonético, se utilizó Tcl, el cual permite mantener la consistencia en las herramientas del *toolkit* y manejar los módulos de sus tecnologías centrales. De esta manera se implanta una extensión del *toolkit* sin tener que cambiarlo.

4.5 Fase de entrenamiento

Esta fase tiene como objetivo que el clasificador “aprenda” las características esenciales de aquello que desea reconocer. Por lo que es importante que los datos empleados en esta fase sean representativos. Durante el proceso de entrenamiento, como se mencionó anteriormente, los pesos se van ajustando gradualmente hasta encontrar aquellos que generen la salida deseada.

Para el desarrollo de la fase de entrenamiento, se corrieron *scripts* de Tcl basados en la estructura del *toolkit*. En cuanto a la organización de los datos, el *toolkit* tiene un directorio de datos con dos subdirectorios: uno de corpus y otro de reconocedores.

El proceso comprendió los siguientes pasos:

- Crear el archivo de *corpora* y el archivo de *files.*
- Crear los archivos de parts y de vocab.
- Correr los *scripts* de entrenamiento.

4.5.1 Crear el archivo de *corpora* y el archivo de *files*

El archivo de *corpora* contiene una lista maestra de cada *corpus,* y para cada *corpus* la localización y formato de sus archivos. No hay manera automática de generar este archivo, pero es fácil de modificar a mano. El mismo archivo de *corpora* debe ser usado para todas las tareas de entrenamiento.

El archivo de *files* contiene la ubicación de los archivos necesarios para entrenar el sistema. Incluye para cada frase del *corpus* la ubicación del archivo correspondiente a la señal de voz (.wav), el archivo de transcripciones a nivel texto (.txt), el archivo de transcripciones a nivel palabra (.wrd), el archivo de transcripciones a nivel fonema (.phn), y el archivo de categorías (.cat).

Para crear este archivo se ejecuta el *script* mk_files de la siguiente manera:

mk_files.tcl -files tele.files -user 300 -corpus teléfono	
-files	nombre del archivo de salida
-user	indica el rango de usuarios a incluir
-corpus	nombre del corpus del que se desea el archivo tipo files

Ejemplo 4.1 Linea de comando para crear el archivo .files

A continuación se da un ejemplo de como se vería un archivo de *files*:

```
/data/corpora/telefono/speechfiles/300/u300.s1.wav
/data/corpora/telefono/transcriptions/300/u300.s1.phn
/data/corpora/telefono/transcriptions/300/u300.s1.cat
/data/corpora/telefono/transcriptions/300/u300.s1.txt
/data/corpora/telefono/transcriptions/300/u300.s1.wrd
/data/corpora/telefono/speechfiles/301/u301.s1.wav
/data/corpora/telefono/transcriptions/301/u301.s1.phn
/data/corpora/telefono/transcriptions/301/u301.s1.cat
/data/corpora/telefono/transcriptions/301/u301.s1.txt
/data/corpora/telefono/transcriptions/301/u301.s1.wrd
…
…
```

Ejemplo 4.2 Formato de un archivo tipo files

4.5.2 Crear el archivo de parts y de vocab

Estos dos archivos son necesarios para empezar con la etapa de entrenamiento. El archivo .parts especifica para cada uno de los fonemas el número de partes en que debe ser dividido. También contiene los grupos de contexto a usar. El archivo .vocab contiene el vocabulario, pronunciaciones, y la gramática para la tarea. Este también debe ser creado a mano. A continuación se dan los formatos clásicos que deben tener estos dos archivos.

```
.pau     1 ;          i_h      1 ;          s        1 ;
.bn      1 ;          i_fp     1 ;          s_v      1 ;
.br      1 ;          i_x      1 ;          t        1 ;
.ln      1 ;          x        1 ;          tc       1 ;
.unk     1 ;          x_fp     1 ;          tS       1 ;
a        1 ;          k        1 ;          tSc      1 ;
a_h      1 ;          kc       1 ;          u        1 ;
a_fp     1 ;          l        1 ;          u_h      1 ;
a_x      1 ;          dZ       1 ;          u_fp     1 ;
b        1 ;          dZc      1 ;          u_x      1 ;
bc       1 ;          m        1 ;          w        1 ;
d        1 ;          m_fp     1 ;          V        1 ;
D        1 ;          n        1 ;          ae       1 ;
dc       1 ;          N        1 ;          ai       1 ;
e        1 ;          nj       1 ;          ei       1 ;
e_h      1 ;          o        1 ;          ia       1 ;
e_fp     1 ;          o_h      1 ;          ie       1 ;
e_x      1 ;          o_x      1 ;          io       1 ;
f        1 ;          p        1 ;          oi       1 ;
g        1 ;          pc       1 ;          wa       1 ;
G        1 ;          r        1 ;          we       1 ;
gc       1 ;          rr       1 ;
i        1 ;          rZ       1 ;
```

Ejemplo 4.3 Formato de un archivo .parts

Y el formato del archivo de vocabulario es de la forma de la tabla 4.1, son dos columnas, la primera con los fonemas del vocabulario, y la segunda con sus alófonos, además de una gramática. Para una mayor referencia ver apéndice A, archivos tele.vocab y tele.parts.

Aquí hay que resaltar algunos detalles importantes de este experimento base, como lo son:

1. En este experimento no existen clases generales de fonemas, debido a que se pretende que cada fonema sea totalmente independiente de cualquier otro.
2. Cada uno de los fonemas se dividió en una parte debido a que es el experimento básico.
3. No existen ningún tipo de mapeo.
4. Las “palabras” en el archivo vocab son fonemas, y sus “pronunciaciones” son alófonos; osea, se usa la infraestructura diseñada para reconocimiento a nivel de palabras para reconocimiento a nivel de fonemas.

4.5.3 Correr los *scripts* de entrenamiento

En esta etapa se corren los siguientes *scripts* de Tcl:

- make_recognizer.tcl encuentra los archivos para entrenamiento y determina las categorías que serán clasificadas por el reconocedor.
- gen_catfiles.tcl toma la lista de los archivos para entrenamiento y crea etiquetas alineadas con el tiempo de las categorías a entrenar. En este caso de un reconocimiento independiente del contexto, las categorías son simplemente las etiquetas contenidas en las transcripciones a nivel de fonema, osea en los archivos .phn.
- revise_desc.tcl asegura que para cada categoría, hay suficientes ejemplos para entrenar. También agrega límites de duraciones en los archivos .desc y .olddesc.
- genvec.tcl asocia los vectores de cada *frame* con su categoría, preparando los datos para la etapa de entrenamiento.
- train_and_test_nets.tcl entrena las redes neuronales usando el archivo de vectores *name.train.vec*. Este programa crea un archivo de entrenamiento representando la

red después de cada iteración. Luego evalua cada red sobre los datos de desarrollo.

El proceso de entrenamiento es iterativo; es decir, la red será entrenada cierto número de veces, el cual puede variar según sea necesario. Para está tesis se decidió tomar el *default* de treinta iteraciones, observando que después de la treintava iteración la curva de reconocimiento tendía a bajar. Cada iteración se va guardando en un archivo de salida llamado *nnet.*

4.6 Fase de desarrollo

La etapa de desarrollo consiste en determinar cuál fue la red que obtuvo el desempeño más alto. Para esto se evalúa el nivel de error alcanzado en cada iteración haciendo uso de los vectores de características generados para esta etapa. Una de las técnicas, y la utilizada por el *toolkit*, consiste en determinar el nivel de reconocimiento en términos del grado de error generado [Cole96]. Para realizar este cálculo, se aplica la siguiente fórmula:

$$E = \frac{S + I + D}{N} * 100$$

En donde N es el número total de palabras en el conjunto de prueba, S es el número de substituciones, I es el número de inserciones y D el número de supresiones. Este método es aplicado a nivel unidades y frases. Al terminar esta fase se obtiene la red que tiene el mejor desempeño para utilizarla en la etapa final.

4.6.1 Script de conversión

Los datos de desarrollo sobre los cuales queremos evaluar el desempeño de cada red están etiquetados no a nivel de fonemas, si no a un nivel más bajo aún. Por ejemplo la “t” está etiquetada a {tc t} cuando nos interesa reconocer “t fonema” ó t_f. Por otro lado no se quiere penalizar por confundir alófonos {bc b} vs. {V} ó {we} vs. {ue}. De hecho la distinción es muy subjetiva aún para las personas que hicieron las etiquetas.

Para solucionar esta problemática se hizo un *script* convierte.tcl (ver apéndice B) el cual utiliza información de los archivos .wrd y .phn, los recorre uno por uno y va mapeando todas las pronunciaciones a unidades de fonemas. A continuación se muestra una tabla de estas dos representaciones.

Nivel	Representación
Textual	hola buen dia
Fonemas	O_F L_F A_F B_F U_F E_F N_F D_F I_F A_F
Alófonos	o l a bc b u e n dc d i a

Tabla 4.3 Niveles de representación

Para realizar la conversión a unidades fonéticas, este *script* usaba un conjunto de reglas. Estas reglas modelaban al conjunto de fonemas y algunos diacríticos. (ver apéndice B, archivo reglas.phones).

Una vez realizadas estas dos tareas se llevó a cabo la etapa de evaluación. Esta consistió en la comparación de la salida del reconocedor con los archivos .fone, los cuales son los archivos referencia.

4.7 Resultados con CSLU *toolkit*

Al revisar manualmente la calculación de error calculado por el CSLU *toolkit,* se notó que éste no estaba evaluando correctamente. Por ejemplo, si al principo de la frase había una inserción por el reconocedor, automaticamente todo lo subsecuente no coincidía y lo tomaba como substiticiones, aún cuando se observaba que realidad e hipótesis coincidian. Por ello, se optó por cambiar el software de evaluación y usar un programa hecho por el *National Institute of Standards and Technology* (NIST)[1].

4.8 Nist

El software de NIST, antes de calcular el porcentaje de error, corre un algoritmo de alineación de cadenas para minimizar el número de diferencias (substituciones, eliminaciones, e inserciones) entre los dos. El paquete de software fue desarrollado y probado usando el sistema operativo UNIX 4.3. Este software tiene dos propósitos: Primero, alentar a los investigadores quienes están reportando resultados empíricos a usar medidas estadísticas para resumir sus mejoras y conclusiones; y segundo, proveer una manera estándar para medir el porcentaje de reconocimiento, asegurando que las diferencias en resultados publicados por diferentes grupos de investigación son debido a sus reconocedores y no a sus algoritmos para calcular los resultados.

Estas herramientas de software y corpus de voz han sido desarrolladas para el uso de la comunidad en general en el área de reconocimiento de voz [Pallett90]. De hecho, los resultados de la tabla 4.2 fueron evaluados con el mismo software.

[1] http://www.itl.nist.gov/div894/894.01/software.htm

De acuerdo al método de evaluación es matemáticamente posible tener una precisión menor que cero, debido a la presencia de errores de inserción. El software de NIST utiliza un archivo de referencia y otro de frases hipotéticas. Ambos archivos deben estar en un formato específico que se muestra a continuación:

```
R_F O_F D_F R_F I_F G_F O_F (300a0009)
T_F E_F R_F E_F S_F (300a0008)
S_F I_F (300a0000)
```

Ejemplo 4.4 Formato de archivos en NIST, frases (rodrigo, teres, si).

Como los archivos de salida wrd_align del CSLU *toolkit* tienen un formato diferente a los que ocupa NIST, fue necesario hacer dos scripts en Tcl que convirtieran de un formato a otro automaticamente. Estos dos scripts son crea_hyp.tcl y crea_ref.tcl (ver apéndice B).

Una vez ya con los archivos en el formato correcto, se procedió a ejecutar el software para el conjunto de desarrollo. Este proceso incluyó los siguientes pasos:

- Alineación
- Scoring

A continuación se describe con más detalle cada uno de estos pasos.

4.8.1 Alineación

Primero, el programa de alineación lee el archivo conteniendo una lista de las sentencias hipotéticas. Entonces, se utiliza un algoritmo de programación dinámica para alinear cada sentencia hipótesis a su transcripción ortográfica. Después de realizar todas las alineaciones, cada par de unidades es etiquetada como reconocida

correctamente o resultado de una inserción, substitución o eliminación. Finalmente, la estructura que contiene todo el sistema de sentencias se escribe en un archivo binario con extensión .ali.

```
id: (304A0012)   #ref = 11
REF:  n_f o_f B_F e_f n_f t_f a_f A_F Ñ_F o_f s_f ***
HYP:  n_f o_f J_F e_f n_f t_f a_f B_F L_F o_f s_f O_F
EVAL:         S                   S   S           I


id: (304A0002)   #ref = 15
REF:  s_f E_F S_F e_f n_f t_f A_F I_F d_f o_f *** S_F a_f *** Ñ_F
o_f s_f
HYP:  s_f *** L_F e_f n_f t_f *** E_F d_f o_f J_F I_F a_f L_F I_F
o_f s_f
EVAL:     D   S               D   S           I   S       I   S
```

Ejemplo 4.5 Alineaciones hechas por NIST

4.8.2 *Scoring*

Una vez hecho el archivo de alineación, el sistema debe medir los niveles de desempeño usando el programa “score”. El programa *score* lee el archivo de alineación, y reporta los resultados por locutor, por frase, y un resumen general de toda la evaluación.

4.9 Resultados finales (con NIST)

El proceso de evaluación con NIST se aplicó a las quince redes o iteraciones del reconocedor fonético y se obtuvieron los siguientes resultados:

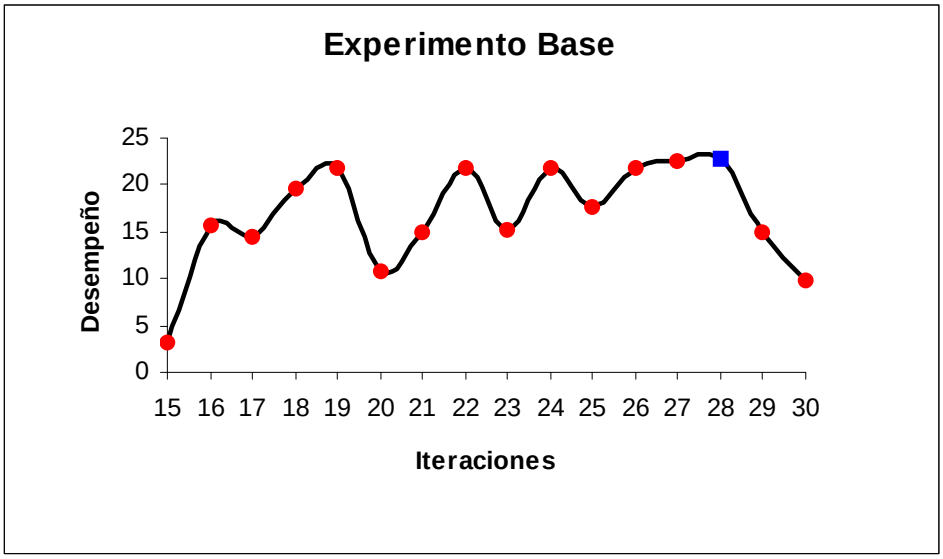

Figura 4.1 Iteraciones del experimento base

Iteración	Desempeño
30	09.89 %
29	14.97 %
28	**22.90 %**
27	22.46 %
26	21.79 %
25	17.67 %
24	21.80 %
23	15.13 %
22	21.83 %
21	14.89 %
20	10.69 %
19	21.75 %
18	19.53 %
17	14.42 %
16	15.65 %
15	03.25 %

Tabla 4.4 Desempeño de las quince redes del experimento base.

En este caso la mejor red fue la número **28**. Para este conjunto de **70** locutores y **1214** frases, el número promedio de frases por locutor fue de **17**. El sistema tiene un porcentaje de reconocimiento de **22.9 %**, un 57.3% de inserciones, un 4.1% de eliminaciones y un 15.7% de substituciones. La desviación estandar del error para todo el sistema fue de 22.7%.

En la siguiente tabla se muestra como sería un reporte detallado en el software de evaluación de NIST:

Loc	# Frs	Corr	Sub	Eli	Ins	Err	S. Err
342	19	85.8	10.2	4.1	38.2	52.4	89.5
359	16	51.6	32.4	16.0	25.0	73.4	87.5
365	18	77.9	20.6	1.4	64.5	86.5	88.9
368	17	70.6	20.9	8.5	48.4	77.8	88.2

Tabla 4.5 Apariencia de un reporte detallado en NIST.